OS Projektas

II dalis

Rytis Petrauskas

Simas Karaliūnas

Procesai

Procesas – tai kokioje nors veiklumo stadijoje esanti programa, kartu su jos esamomis registrų reikšmėmis ir savo kintamaisiais. Veiklumo stadiją apibūdina proceso aprašas – deskriptorius. Šiame deskriptoriuje ir bus laikomi visi procesui reikalingi parametrai.

Proceso aprašas tai dinaminis objektas, kuris gali būti sukurtas ir sunaikintas jau sistemos veikimo eigoje.

Pačius procesus kuria procesai, tai reiškia, jog egzistuos vienas pagrindinis procesas, kuris sukurs visus kitus, jį pavadinsime StartStop.

Tipai

Procesai yra skirstomi į vartotojiškus ir sisteminius. Sisteminiai procesų paskirtis – aptarnauti vartotojiškus, tuo tarpu vartotojiškų – vykdyti programą.

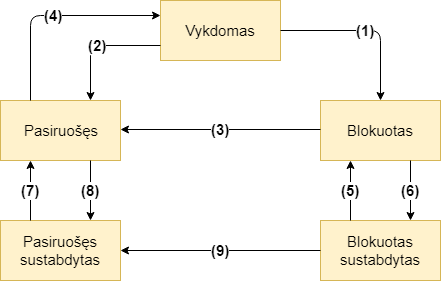
Sisteminiai procesai sukuriami ir sunaikinami kartu su sistema.

Vartotojo procesai yra sukuriami sisteminių procesų pagalba. Kartu jei reikia gali būti sukurta ir daugiau sisteminių procesų aptarnaujančių tą vartotojo procesą.

Būsenos:

Kiekvienas procesas gali gauti procesorių tik turėdamas visus kitus jam reikalingus resursus. Procesas gavęs procesorių tampa vykdomu ir išlieka šioje būsenoje tol, kol sistemoje neįvyksta pertraukimas arba jam prireikia papildomo resurso (pavyzdžiui įvedimo iš klaviatūros). Tokiu atveju procesas blokuojasi. Tačiau net ir nepareikalavus resurso iš proceso gali būti atimamas procesorius, pavyzdžiui, vien dėl to, kad per ilgai dirbo. Tai yra laikoma visiškai skirtinga būsena nei blokavimas dėl resurso.

Galimos proceso būsenos:

* Vykdomas – turi procesorių;
* Blokuotas – prašo resurso (išskyrus procesorių);
* Pasiruošęs – turi visus reikiamus resursus, išskyrus procesorių
* Sustabdytas – kito proceso sustabdytas procesas.

*Diagrama vaizduojanti sąryšį tarp procesų būsenų*

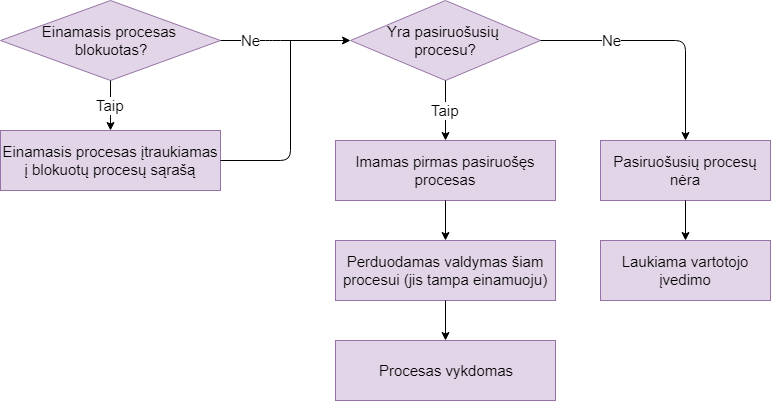
Egzistuoja devyni perėjimai:

1. Vykdomas procesas blokuojasi jam prašant ir negavus resurso
2. Vykdomas procesas tampa pasiruošusiu atėmus iš jo procesorių dėl kokios nors priežasties
3. Blokuotas procesas tampa pasiruošusiu, kai yra suteikiamas reikalingas resursas.
4. Pasiruošę procesai varžosi dėl procesoriaus. Gavęs procesorių procesas tampa vykdomu
5. Procesas gali tapti sustabdytu blokuotu, jei einamasis procesas jį sustabdo kai jis jau ir taip yra blokuotas.
6. Procesas tampa blokuotu iš blokuoto sustabdytu, jei einamasis procesas nuima būseną sustabdytas.
7. Procesas gali tapti pasiruošusiu sustabdytu, jei einamasis procesas jį sustabdo, kai jis yra pasiruošęs.
8. Procesas tampa pasiruošusiu iš pasiruošusio sustabdyto, jei einamasis procesas nuima būseną sustabdytas.
9. Procesas tampa pasiruošusiu sustabdytu iš blokuoto sustabdyto jei procesui yra suteikiamas jam reikalingas resursas.

Planuotojas:

Planuotojas – centrinio procesoriaus resurso paskirstytojas. Jo užduotis – atimti procesorių iš vykdomojo proceso, peržvelgti pasiruošusių sąrašą ir išrinkus tinkamiausią procesą, perduoti jam procesorių. Planuotojo tikslai:

* Užtikrinti, kad kiekvienas procesas pagristą laiko tarpą turėtų procesorių;
* Maksimaliai išnaudoti procesorių
* Sumažinti atsakymo laiką vartotojams iki minimumo

Planuotojo algoritmas veikia prioritetų modeliu. Kiekvienas procesas turi prioritetą, nusakantį jo svarbą. Šis prioritetas bus apibrėžta skaitine reikšme nuo 1 iki 100, laikant jog 1 – tai žemiausia proceso svarba, tuo tarpu 100 – pati auksčiausia. Procesas turintis aukščiausia prioriteta bus vykdomas pirmiausiai.

*Diagrama vaizduojanti planuotojo veiksmų seką*

Procesų primityvai:

Procesų primityvų paskirtis - pateikti vienodą ir paprastą vartotojo sąsają darbui su procesais. Kiekvieno primityvo programos gale yra iškviečiamas planuotojas. Išvis yra 4 procesų primityvai:

1. **Kurti procesą.** Šiam primityvui perduodama nuoroda į jo tėvą, jo pradinė būsena, prioritetas, perduodamų elementų sąrašas ir išorinis vardas. Pačio primityvo viduje vyksta proceso kuriamasis darbas. Jis yra registruojamas bendrame procesų sąraše, tėvo-sūnų sąraše, skaičiuojamas vidinis identifikacijos numeris, sukuriamas jo vaikų procesų sąrašas (tuščias), sukurtų resursų sąrašas.

1. **Naikinti procesą.** Pradedama naikinti proceso sukurtus resursus ir vaikus. Vėliau išmetamas iš tėvo sukurtų procesų sąrašo. Toliau išmetamas iš bendro procesų sąrašo ir, jei reikia, iš pasiruošusių procesų sąrašo. Galiausiai naikinami visi jam perduoti resursai ir proceso deskriptorius yra sunaikinamas.
2. **Stabdyti procesą.** Keičiama proceso būsena iš blokuotos į blokuotą sustabdytą arba iš pasiruošusios į pasiruošusią sustabdytą. Einamasis procesas stabdomas tampa pasiruošusiu sustabdytu.
3. **Aktyvuoti procesą.** Keičiama proceso būsena iš blokuotos sustabdytos į blokuotą, ar pasiruošusios sustabdytos į pasiruošusią.

Resursai

Resursas yra tai, dėl ko varžosi procesai. Tam, kad procesas galėtų vykdyti savo užduotį (kodą), jam reikalingas resursas. Esant resursų trūkumui, procesai blokuojasi. Procesas sulaukęs reikiamo resurso tampa pasiruošęs. Kiekvienas resursas turi laukiančių procesų sąrašą , kuris gali būti ir tuščias. Į resurso laukiančių procesų sąrašą yra dedami procesai, kurie prašė resurso, bet jo vis dar negavo

Resursus galima skirstyti į:

* **Statinius.** Šie resursai gali būti naudojami **daug kartų**. Kuriami sistemos kūrimo metu. Tai mašinos resursai, tokie kaip procesorius, atmintis ar kiti resursai, kurie sistemos veikimo metu nėra naikinami. Šie resursai būna laisvi, kai jie nėra naudojami nei vieno proceso. Nelaisvi, kai juos naudoja vienas ar keli procesai. Tačiau verta pabrėžti, kad ne visi resursai gali būti skaidomi, t.y. naudojami kelių procesų vienu metu (pvz: klaviatūra, ekranas).
* **Dinaminius.** Šie resursai gali būti panaudoti tik **vieną kartą**. Kuriami ir naikinami sistemos darbo metu. Šie resursai naudojami kaip pranešimai. Kartu su jais gali ateiti naudinga informacija.

Resursų primityvai:

Resursas turi keturis primityvus:

1. **Kurti resursą.** Resursus kuria tik procesas. Resurso kūrimo metu perduodami kaip parametrai: nuoroda į proceso kūrėją, resurso išorinis vardas. Resursas kūrimo metu yra: pridedamas prie bendro resursų sąrašo, pridedamas prie tėvo suskurtų resursų sąrašo, jam priskiriamas unikalus vidinis vardas, sukuriamas resurso elementų sąrašas ir sukuriamas laukiančių procesų sąrašas.
2. **Naikinti resursą.** Resurso deskriptorius išmetamas iš jo tėvo sukurtų resursų sąrašo, naikinamas jo elementų sąrašas, atblokuojami procesai, laukiantys šio resurso, išmetamas iš bendro resursų sąrašo, ir, galiausiai naikinamas pats deskriptorius.
3. **Prašyti resurso.** Šį primityvą kartu su primityvu “atlaisvinti resursą” procesai naudoja labai dažnai. Procesas, iškvietęs šį primityvą, yra užblokuojamas ir įtraukiamas į to resurso laukiančių procesų sąrašą. Sekantis šio primityvo žingsnis yra kviesti resurso paskirstytoją.
4. **Atlaisvinti resursą.** Šį primityvą kviečia procesas, kuris nori atlaisvinti jam nereikalingą resursą arba tiesiog perduoti pranešimą ar informaciją kitam procesui. Resurso elementas, primityvui perduotas kaip funkcijos parametras, yra pridedamas prie resurso elementų sąrašo. Šio primityvo pabaigoje yra kviečiamas resursų paskirstytojas.

|  |  |  |
| --- | --- | --- |
| Resursas | Kūrėjas | Naudotojai |
| supervisorMem | StartStop | ReadFromHDD |
| userMem | StartStop | Loader |
| keyboardCh | StartStop | Input |
| screenCh | StartStop | Output |
| outString | StartStop | Output |
| sysInterrupt | StartStop | Interrupt |
| sharedMem | StartStop | JobGovernor |
| codeInSM | ReadFromHDD | Checker |
| codeChecked | Checker | Loader |
| codeReady | Loader | MainProc |
| paging | Loader | JobGovernor |
| userInput | Input | ReadFromHDD, JobGovernor |
| osEnd | Input | StartStop |

Paskirstytojai

Kaip procesorius yra skirstomas planuotojo, taip ir kiekvienas resursas yra skirstomas tam tikro paskirstytojo. Procesas, prašydamas resurso, arba norėdamas jį atlaisvinti kreipiasi į resursų paskirstytoją, kuris jį aptarnauja. Paskirstytojo paskirtis - suteikti paprašytą elementų kiekį procesui.

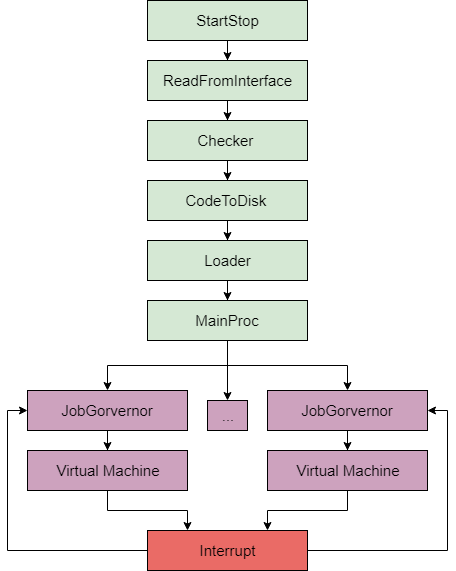
Tačiau, statiniai resursai gali būti panaudojami daug kartų, dėl jų konkuruoja daug procesų. Tad tam, kad išspręstume šią problemą yra naudojamas laukiančių procesų sąrašas, kurį jau minėjome anksčiau. Resursų paskirstytojas eina per sąrašą, kuris veikia eilės principu (procesas, kuris anksčiausiai paprašė resurso, gauna jį pirmiau). Procesui gavus savo norimą resurso elementą, paskirstytojas išbraukia šį procesą iš laukiančiųjų eilės. Paskirstytojo pabaigoje yra iškviečiamas planuotojas.

Procesų paketas

Naudosime šiuos procesus:

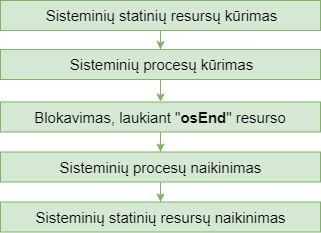
* **StartStop** – šakninis procesas, sukuriantis bei naikinantis sisteminius procesus ir resursus.
* **ReadFromInterface** – užduoties nuskaitymo iš įvedimo srauto procesas.
* **CodeToDisk** – užduotis perkelia vartotojo programą iš supervizorinės atminties į išorinį diską.
* **Checker** – užduoties programos suskaidymas blokais ir jų organizavimas kaip resursus.
* **Loader** – iš išorinės atminties duomenys perkeliami į vartotojo atmintį.
* **MainProc** – procesas, valdantis JobGorvernor procesus.
* **InputOutput** – procesas darbui su išvedimu bei įvedimu.
* **JobGorvernor** – virtualios mašinos proceso tėvas, valdantis virtualios mašinos proceso darbą.
* **VirtualMachine** – procesas, atsakantis už vartotojiškos programos vykdymą.
* **Interrupt** – procesas, apdorojantis virtualios mašinos pertraukimą sukėlusią situaciją.

Beveik visi procesai yra sukuriami sistemos darbo pradžioje proceso StartStop. StartStop nekuria tik 2 procesų – JobGorvernor (kiekvienai naujai vartotojo užduočiai MainProc kuria po naują procesą JobGorvernor) ir VirtualMachine, kurį kuria JobGorvernor.

*Procesų paketo schema iš vartotojo užduoties pusės*

StartStop

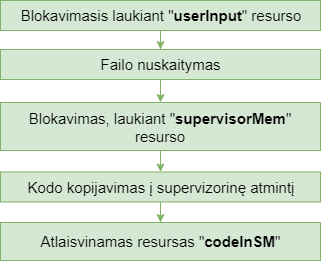
**StartStop** – šakninis procesas, atsakingas už sistemos darbo pradžią ir pabaigą. Įjungus kompiuterį pasileidžia automatiškai ir yra atsakingas už sisteminių procesų ir resursų kūrimus.



*Proceso StartStop veikimo schema*

ReadFromHDD

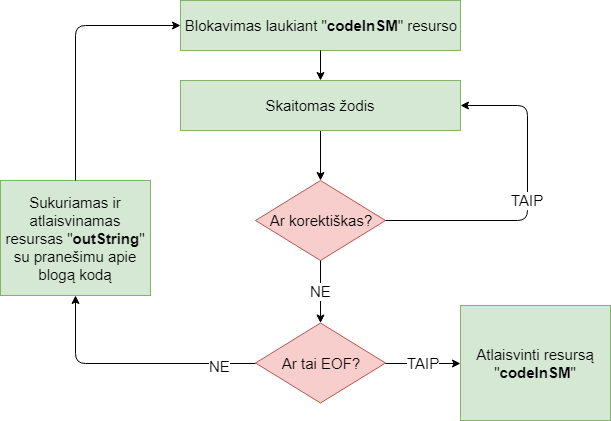
**ReadFromHDD** – procesas kopijuojantis duomenis iš kietojo disko į supervizorinę ir atiduodantis tolesniam apdorojimui, kurį atlieka Checker. Šis procesas yra kuriamas ir naikinamas StartStop proceso. Procesas laukia įvedimo srauto iš vartotojo, šiuo atveju, failo pavadinimo kuriame yra programa.



*proceso ReadFromHDD veikimo schema*

Checker

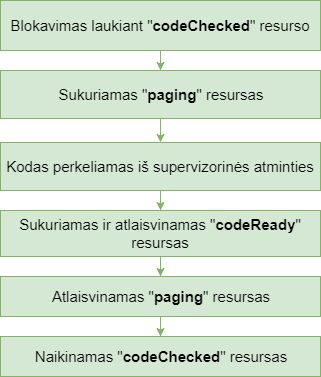
**Checker** – procesas, tikrinantis nuskaityto kodo korektiškumą



*Proceso Checker veikimo schema*

Loader

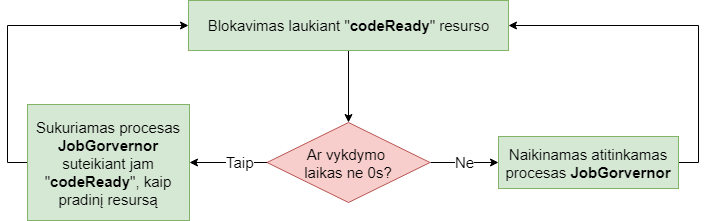
**Loader** – sisteminis procesas, atsakingas už duomenų perkelima iš supervizorinės atminties į vartotojo atminty ir puslapiavimo paruošimą.



*Precoso Loader veikimo schema*

MainProc

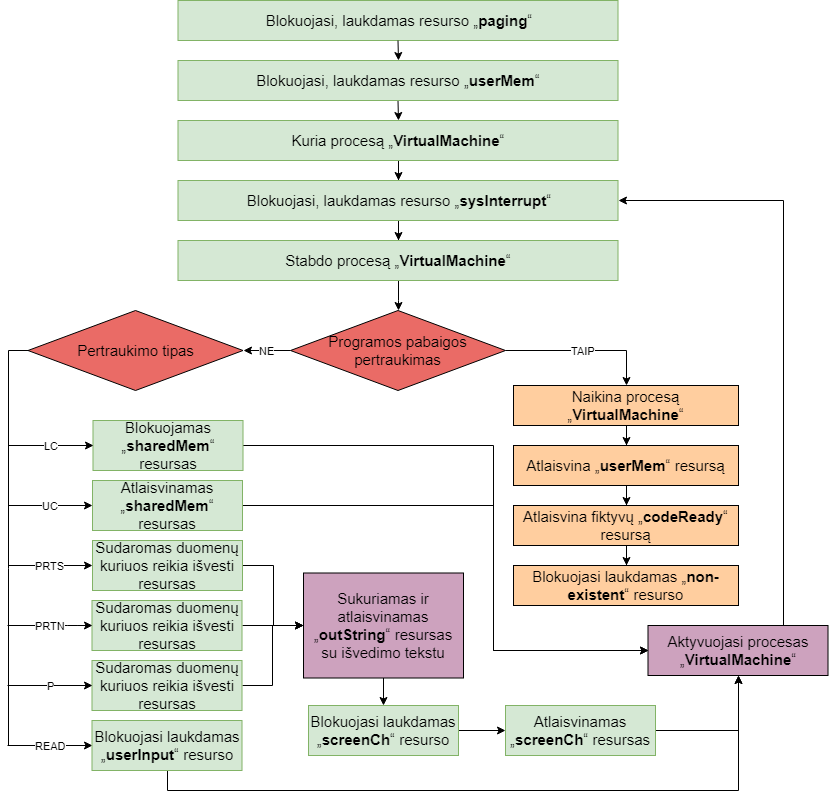
**MainProc** – procesas, kuriantis ir naikinantis procesus JobGovernor



*Proceso MainProc veikimo schema*

JobGovernor

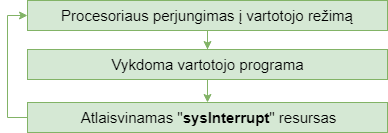
**JobGovernor** – valdo vieną procesą VirtualMachine



*Proceso JobGovernor veikimo schema*

VirtualMachine

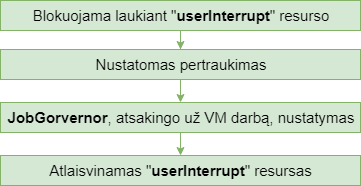
**VirtualMachine** – proceso paskirtis vykdyti vartotojo programas. Jį kuria ir naikina JobGovernor procesai.



*Proceso VirtualMachine veikimo schema*

Interrupt

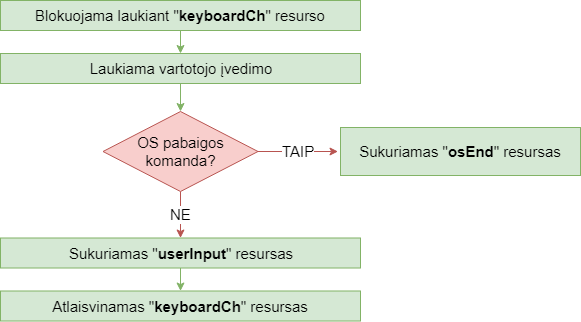
**Interrupt** – Procesas, skirtas apdoroti kylančius pertraukimus.



*Proceso Interrupt veikimo schema*

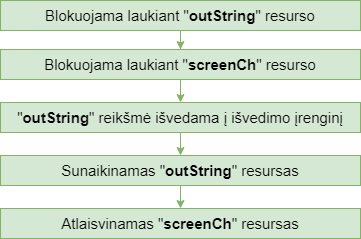
Input

**Input** – Procesas, atsakingas už duomenų įvedimą iš įvedimo įrenginio (klaviatūros).



Output

**Output** – Procesas, atsakingas už duomenų išvedimą į išvedimo įrenginį (ekraną).



*Output*